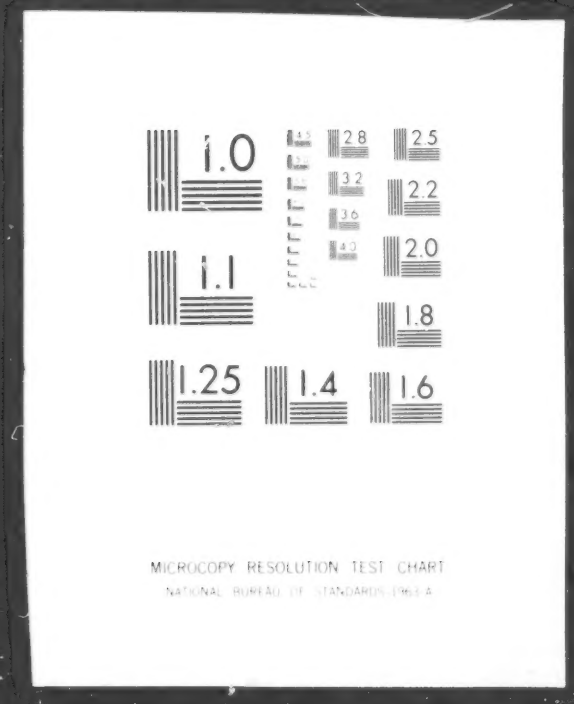
1.0
1.1
1.25
1.4
1.6
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.8
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ED193032

U.S. DEPARTMENT OF HEALTH
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

PERMISSION TO REPRODUCE THIS
MATERIAL HAS BEEN GRANTED BY
J.R. Hartley

TO THE EDUCATIONAL RESOURCES
INFORMATION CENTER (ERIC)

SE 032 876

Occasional Publication Number 2

USING THE COMPUTER TO STUDY AND ASSIST THE LEARNING OF MATHEMATICS

J R HARTLEY

1980

Included in the Proceedings of the
British Society for the Psychology
of Learning Mathematics Conference.
University of Nottingham
3-4th January 1980

USING THE COMPUTER TO STUDY AND ASSIST THE LEARNING OF MATHEMATICS

J. R. Hartley
Computer Based Learning Project
The University of Leeds

ABSTRACT

Computer Assisted Learning (CAL) can be considered as an exercise in designing languages so that appropriate educational interactions can take place at the terminal. Conventions (e.g. language commands, permitted types of response) have to be specified so that the student can use and learn from the programs, and these conventions, together with the material on which they operate, determine the range of cognitive and learning activities open to him. Educational objectives, factors known to influence the particular types of learning they embrace, and the characteristics of the electronic computer should determine what these conventions - and hence the designs of the materials - should be.

Three general types of learning have been distinguished, namely the acquisition of concepts, the learning of relations and rules, and problem solving. In acquisition, the representational mode of the material (e.g. diagrams, verbal statements, or symbolic descriptions) and its semantic structure influence the encoding activity of the learner. This is the important phase in learning, and can be controlled by embedding tasks within the material which influence the ways students use previous knowledge and organise their semantic structures. In mathematics the student has to learn how to make symbolic descriptions meaningful. Thus CAL developments have concentrated on the use of educational programming languages (e.g. LOGO) for illustration and reasoning. Other work has provided the student with commands to give him selection-control over the type and sequence of educational material. Further developments have tried to adapt such structuring to the learner's preferred styles of working and thinking.

In rule and relation learning, the influence of feedback, its control through task difficulty, and the same-context/varied-context menus of practice are important. To handle these variables in a knowledgeable manner requires the program to have a representation of the topic area (i.e. to be able to work out the learning tasks in expert mode) and knowledge of the student (i.e. how he undertakes the tasks). Some examples of methods for setting up and using such models are given. The way that computer based mathematical models can interplay with mathematical reasoning to encourage the student to hypothesise and conjecture about their underlying relations is also outlined.

In problem-solving, computer orientated work has concentrated on trying to describe in a detailed and reliable manner the problem-solving performances of subjects. These information processing descriptions (set in the form of production-rule types of computer programs) can be used to examine how students perceive/formulate problems, and how they develop strategies, and so illuminate individual differences.

Finally, the assimilation of CAL within the conventional classroom will not be easy. The former must emphasise individualisation in learning, but the latter, for economic reasons, has to restrict the differing conditions under which students learn. Languages which allow teachers to set up their own CAL schemes, to develop material once its structure and design have been decided, and to access individual and class records, will be valuable, even necessary, if coherent man-machine teaching systems are to be developed.

1. INTRODUCTION

The title may be judged optimistic on three counts. First, although there has been a greatly increased interest in the teaching capabilities of the electronic computer over the last decade, only a small proportion of this research and development work is set in the learning of mathematics as distinct from its applications. Second, even these studies tend to lack sufficient rationale, and do not systematically relate those factors influencing learning to teaching program designs. Thus, experience is hard to generalise, and the undoubted energies of research workers are

only loosely harnessed to a common core of learning and technical problems. Third, there are few developmental projects which are firmly rooted in the classroom, and of sufficient scale to tackle the considerable problems of assimilating a new technology within conventional teaching schemes. That being said, there have been several noteworthy achievements, and the purpose of this paper is to comment on these and indirectly invite more practitioners to become involved with computer assisted learning (CAL) in mathematics.

Within the teaching and subject matter of mathematics, the characteristics of the electronic computer should determine its educational functions. For example, the computer can quickly carry out numerical, algebraic and logical operations. If the relationships underlying a subject area can be set out in these terms, computer programs can illustrate them by allowing the user a command language to express these relations, and to control input values and the consequent output displays in numerical or graphical forms. Topics in mathematics can, indeed often must be represented in these symbolic, structured terms, and this formality can cause learning difficulties for students particularly those applying mathematical techniques in the biological and social sciences. Typically these students are varied in the mathematics experience and conceptualisations they bring to the tasks. Hence readily displayed illustrations of concepts (e.g. standard error of estimate) and relations should prove both stimulating and useful - and are often difficult to produce by other means. It is worth comment that through visual display units and graphics terminals, the output can be quickly and attractively displayed; note also that the command languages and simulation packages can be teacher controlled for class demonstration (using CCTV monitors connected to the terminal) or pupil controlled through individual exercises or mini-projects worked at the terminal.

Although the grammars underlying verbal statements are not as precise or as well-regulated as those of algebras or logics, the computer can store large amounts of (labelled) information and display/print these textual messages. User responses, e.g. those typed in textual form at the keyword, can be evaluated through program matching routines and anticipated alternatives supplied by the program author. Thus question-answer discourse can proceed. Of course, there have to be limitations on

these techniques which use pre-stored material - limitations on size of step or task, and/or on types of statements or commands the student can use. On the other hand such programs stimulate and control student learning activity. Indeed, whether simulation packages, command languages or textual programs are used, they are driven by the student's responses at the terminal. CAL is an active form of learning but note that the learner's mental processing is prescribed by the type of user language or user package which has been designed. The student can be asked to paraphrase, relate, summarise, give examples, provide explanations, question goals (why) and methods (how). The initiative can move between the teaching program and the user, so that he may question or command. Hence programs can be designed for illustration or acquiring concepts, for learner/program controlled tutoring, for rule-learning and practice, and for problem solving. The general point is that through these facilities the student's cognitive processing is stimulated and regulated; this determines the cognitive structures which are developed and the learning outcomes which result. Designing languages to allow these educational interactions to take place at the terminal is the essence of CAL.

Since the student's responses can be evaluated and stored, computer programs can keep individual records and make inferences about his knowledge and capabilities, perhaps even setting up models of his cognitive processing. This data can be used in a variety of ways. First it can regulate the feedback which is given to a response. It is one of the most documented principles in educational psychology that feedback contingent upon a response is a necessary requirement for learning. Clearly this can be provided in several forms. In simulation packages and educational programming languages, the user's commands and instructions are carried out if they are syntactically correct. The resulting output must be matched against the user's intentions and expectations, i.e. it is a form of self-evaluation. Other tutorial types of CAL programs give the feedback as direct comment. How much information and of what type determines how the learner will locate and correct his errors. The stored performance data is also used by the program to determine a student's route through the material. For example, it may decide on the type of task, its difficulty and the type of help facilities which are made available to the learner.

A second use of student records is to enable the teacher to monitor the progress of a class within a particular subject area. Thus he can decide which pupils need extra help, or if further supporting teaching needs to be given to the class as a whole. A third function of performance data is to provide comment to the student on particular errors or areas of weakness and so give guidance for further study. Such developments require languages or command sets to be designed so that the record system can be interrogated by the user and by the teacher.

These comments point to the potential advantages, but also the limitations of CAL. It can enlarge, regulate and guide student learning activity, and through the interactions at the terminal can make the pupil's mental processing more transparent. But although much attention is given to language and program designs, the communication between user and program is necessarily limited to pre-stored anticipated alternative responses and/or limited command sets and/or operational statements which must follow a specified syntax. These characteristics of CAL contrast with and indeed complement those of the human teacher, who, though not able to calculate quickly or service many pupils individually, can communicate easily and intelligently in ways which a computer program cannot rival. Thus the aim is to design and implement a collaborative man-machine teaching system.

Notwithstanding this aim, there are bound to be difficulties in assimilating a new technology (which emphasises the individual nature of learning) within the conventional techniques and organisational schemes of classroom teaching. To help this assimilation CAL systems must (i) be capable of being used in differing ways to suit particular classes, (ii) have record handling languages/programs to help their successful management and (iii) provide methods (which in practice means ultra-high-level languages) to allow teachers without specialist training in computing to design and perhaps produce material to suit their particular objectives.

Within this general framework some CAL research and developmental studies in mathematics and its applications will be reviewed. Comments will attempt to focus on (i) the models of learning and/or teaching on which the programs are based (three general types of learning have been distinguished, namely, the acquisition of concepts, the learning of

relations and rules, and problem solving), (ii) techniques of program/language design and (iii) how materials are used in order to achieve particular learning outcomes. The paper will conclude with some observations on the present requirements and future directions of computer assisted learning.

2. THE ACQUISITION OF MATHEMATICAL CONCEPTS

2.1 Some experiments on comprehension and recall

Before designing CAL programs to aid the acquisition of mathematical concepts it is necessary to delineate presentation and instructional variables which influence the quality of a student's recall and comprehension of material. What type of knowledge structures does he develop and store? What is the range of individual differences? In an attempt to clarify these questions a series of three experiments was carried out with Polytechnic students studying topics in Statistics.

The material for the first experiment concerned the definitions and properties of the binomial probability distribution. The content of approximately eighty sentences was taken from lecture transcripts and standard textbooks, and arranged in eight paragraphs. The experimental variables were presentation mode and the semantic structure of the material. Presentation of the content was given by audio tape or by visual slides proceeding at the same rate. The structure of the topic was represented as a semantic network following techniques deriving from the LNR language of Lindsay, Rumelhart and Norman. Using different arrangements of the same sentences, low-structure and high-structure versions of the content were prepared from the semantic network. Indexes of structure were calculated for these versions and the validity of the estimates was checked against the ratings of four independent judges. Thus each of the four treatment groups received the content in taped or slide form, in either high or low structured versions. In fact each presented content had two structures, the arrangement of semantic neighbours determining the high or low order, and a serial order deriving from neighbouring sentences which were adjacent in the actual presentation. Approximately one hundred students took part in the experiment; they were rated on previous mathematics

experience and allocated to the four independent treatment groups following a stratified random sampling procedure.

Each group experienced the same sequence of events. The material was presented, and subjects asked to write down what they could recall. Correct order was not essential, accuracy was desirable but the sense of the passage was most important. The material was then presented again, followed by a second recall period, after which a third and final presentation-recall sequence was given.

The results showed several features of interest. Of course, for all the treatment groups there was a significant increase in the number of sentence-propositions recalled on the three occasions. On average, 20%, 35% and 45% of the seventy-eight propositions were recalled on the first, second and third occasion respectively. Sentences tended to be recalled in their entirety or not at all, the number of propositions remembered on the first occasion (analogous to students having received a lecture for example) was small, and the increase followed an almost linear pattern which supports a total processing-time hypothesis.

Analysis of variance showed structure to be a main effect, with high-structure groups recalling the most. Although this difference was statistically significant, it amounted to no more than an increase of 5 - 10% in the total of propositions recalled. Of more interest are differences in the sequence of recall. The low-structure group, on the earlier occasions, recalled sentences which tended to cluster round their social neighbours, i.e. the best predictor (measured by conditional probabilities) of recall was whether its serial neighbour had been remembered. On later presentations the recall sequence moved away from this and became successively closer to the high structured form of the text, i.e. the best predictor was if the semantic neighbour of the proposition had been recalled. The interpretation is that it is the stored (and recalled) semantic structure which is progressively developed rather than a continual referencing of the presented text (Gentner 1976). Incidentally, these results were not influenced by mode of presentation which caused little difference in quantity, quality or sequence of remembering between the groups.

The statements written by the students in their recall sequences were marked as accurately recalled (veridical statements) or overgeneralised -

which might be one way learners seek to reduce the processing load. The fact that the proportion of overgeneralised responses did not decrease on successive occasions supports Frederickson's (1975) view that this characteristic is occuring at the encoding rather than at the retrieval stage.

A second experiment examined further the importance of coding activity on comprehension and recall of concepts. The content took parts of probability related to independent and mutually exclusive events; the presentation was in the form of text (a workbook) which varied the instructions to students in order to regulate their cognitive processing. One treatment informed the subjects that the teaching material was being tested for readability and asked them to underline any word which was more than six letters or might prove difficult to spell. Thus their processing was concentrated at a word and syntactic level. Two other treatments were derived from Craik and Lockhart's views (Craik and Lockhart 1972, and Eysenck M. 1978) on how depth and spread of semantic encoding aid recall. A second group of subjects completed questions embedded in the text which required semantic paraphrasing of the material. A third group - at corresponding points in the text - were asked to provide examples of the propositions. Thus they had to interpret the meaning of the statements within their own experience.

The content was identical for all groups (apart from the processing instructions and tasks); study time was also regulated and kept constant for all groups. Over sixty Polytechnic students took part in the experiment. The material was unfamiliar to them and after being graded on previous mathematical experience, the subjects were assigned to one of the three treatment groups following a stratified random sampling procedure. When the workbook had been completed three post-tests were given - a free-recall, a cued-recall and a comprehension test.

The semantic processing treatments (i.e. the paraphrase and construct-an-example groups) were much the superior on all post-tests. Similar results applied to the structure of the free-recall and it was notable that whereas the sematic groups recalled sentences, the other 'syntax' group recorded words and short phrases. Of the former groups the paraphrase treatment produced greater coherence in free recall. The paraphrase and construct-an-example groups were almost equivalent in quantity of material

remembered, but the former was the superior on cued recall, and the latter the best on the comprehension test. The interpretation of this result is that the different processing instructions engaged and developed different cognitive structures (similar to the conditions of the treatment) and this resulted in different learning outcomes. Further, students with higher ratings on previous mathematical experience were superior particularly on cued recall and comprehension; those with less mathematical competence benefited most from the construct-an-example treatment.

Understanding and comprehension are not only influenced by the processing instructions and tasks which are given but by the type of material which is presented. For example, it is of little value asking the student to 'image' or picture' if the content is presented in formal or abstract terms. In order to use previous experience the material and its context must be given in a form (perhaps through diagrams or analogy) which allows links to be constructed and comparisons to be drawn. A third experiment was undertaken to study such interactions. The content, on conditional probability and Bayes' theorem, was unfamiliar to all the Polytechnic students taking part. One treatment developed the subject formally; terms were defined and formulae used to show the relations between concepts. Examples were given but in abstract terms. The material was well structured, presented by workbook, and embedded questions required semantic paraphrase and examples to be worked. A second treatment covered the same content in the same sequence with the same density of embedded questions. The difference was that symbols and a formal approach were not used, but diagrams and everyday terms. Formulae were not given explicitly but the relations developed with reference to diagrams and everyday experience. Approximately sixty students were rated on previous mathematics experience and allocated to one of the two treatment groups following a stratified random sampling procedure. The two treatments had equal study time and two immediate post-tests were given - a cued-recall test and an application (problem solving) test. Finally an 'impossible' but plausible problem was set.

The results were clear-cut. On the cued-recall test some cues were formal and others were taken from the material which used everyday themes. All the content had been covered by both groups, but the recall performances

were clearly in-line with the type of material the students had received, i.e. the formal group were significantly the better when formal cues were given and vice versa. Overall, because of the balance of cues, the composite recall scores showed no significant differences between treatment groups; students with higher ratings of mathematics experience performed significantly better than the others.

On the problems test which used wordings followed in textbooks and examinations (and favoured no one group), the questions could be worked out following either formal or informal methods. Formulae on probability theory and Bayes theorem were provided, and the questions required the student to apply his knowledge in new contexts which went beyond what had been given in the teaching treatments. The results favoured the informal/diagrammatic group who obtained significantly higher scores, but this difference was largely due to the performances of students low in mathematics experience who obtained very much greater benefits from the informal-diagrammatic approach. Incidentally an examination of the working methods of students largely reflected the approach of the treatment group attended; no candidate was successful or able to make appropriate comment on the 'impossible' problem (in which the probabilities summed to a total greater than 1.0). Thus the experiment supports previous work by Mayer and Greeno (1972) and suggests that different representational forms of material interact with processing activities to develop differing cognitive structures. These can relate critically to students previous experience and thus produce different learning outcomes.

In summary, the design of CAL materials to aid the acquisition and comprehension of mathematical concepts should give attention to (i) the semantic and serial structuring of material, (ii) the type of coding activity which is required of the learner and (iii) the representational mode in which the concepts are developed and the ways these can be related to previous experience. Within this framework it is possible to discern two developments for the computer assisted learning of mathematics. First, the use of specially designed educational programming languages to stimulate and guide student activity and mental processing, and second, the structuring of subject matter and the placing of its selection and use under guided student control.

2.2 Using educational programming languages

Over the last few years the computer based learning project at Leeds has developed and used a simulation language which provides the student with a statistical laboratory. The elements are data sets, and the user can type instructions to control their generation, editing and display, and have arithmetical and statistical operations carried out on them. An instruction statement includes a command to which a parameter set is attached. If a data-label for the output is given, the output is stored under that label (otherwise it is printed directly) and further commands can operate on it. Thus elementary programs can be developed, so that population probability distributions can be specified, samples drawn, outputs displayed in numerical or graphical form, arithmetical and statistical operations carried out, and editing functions used to curtail, change, or merge data-lists. A macro-generation facility allows specific parameters, commands and/or data sets to be labelled as variables during the definition phase, and to be replaced by given and specific names and/or values when called at run-time. Thus procedures can be called successively and their outputs compared. The advantage of this facility of a learning laboratory for the illustration and understanding of statistical concepts is the activity it requires of the learner. He must instruct the machine, and the feedback he receives (apart from a syntax check or his instructions) is the output produced by his program. So the student must formulate and sequence his commands and test the output against his intentions.

A much bolder and more wide-ranging example of this approach to learning is provided by Seymour Papert, who with Wallace Feurzeig has developed the programming language LOGO (Papert 1974, 1971b). Papert has taken a major interest in using LOGO to teach mathematics to school-children, or, as he would prefer, to enable pupils to become mathematicians. To Papert the problem of making mathematics make sense to the learner is the problem of making 'formal descriptions' make sense. He would argue that society, through accidents of history, has made implicit choices about its mathematical languages and its formal descriptions and thus has decreed "the mathematical luggage which its citizens should carry". Superimposed on this, teaching methods have been laggardly in using available technology. Thus for Papert, problems in mathematics education will not

be resolved by tinkering with the present system but by using computer technology to develop and extend the curriculum on rational grounds, and by using teaching methods which fully exploit the learner's experience under the new medium.

Perhaps the best illustration of these ideas occurs in the teaching of geometry to young children. In Papert's view, the Euclidean approach gives an axiomatic and logical style to geometry, Descartes' an algebraic representation, while the language LOGO can provide a computational metaphor strongly related to the learner's own experience. The aim is to allow the child to become an active builder of his own intellectual structures and this is strongly influenced by the language which controls the communication with the computer. This language (LOGO) will circumscribe the metaphors, images and ways of thinking which can be used by the student.

The computational analogue of the Euclidean point is a 'turtle' which has two properties - a 'position' and a 'heading'. The turtle can be an elementary moving robot, connected to the computer, which is able to mark out the routes it travels by inking its path on a piece of paper. Alternatively the turtle can be a cursor on a visual display unit's (vdu) screen. Through the language LOGO, the pupil can issue commands to the turtle, e.g. FORWARD 100 (length units), RIGHT-TURN 90 (degree units). These programs instruct the turtle how to perform actions e.g. draw a triangle or a square. By considering the movement of the turtle, the pupil can debug, amend and improve his programs. From simple procedures, more complex programs can develop, e.g. drawing a house may incorporate the procedures for drawing a square and a triangle. The child will realise that the command-values may require to be changed independently of the commands - thus he appreciates the notion of a variable. The pupil may instruct the turtle to move FORWARD x units, RIGHT-TURN through y units and repeat these instructions recursively. By altering x and y and studying the shapes produced the learner may, for example, discover the circle as a limiting case of an infinitely sided polygon. Putting in additions on the FORWARD step length will cause spiralling; using tests to control the sequence of commands will allow shapes to roll along or round themselves. Such programs can be extended or generalised so that a child can develop his notions of the properties of shape and symmetry.

The main aims of this technique are to firmly root the learning within the pupil's experience and to give the learner power to do things and find out about things which he could not do before. Papert describes such learning as being body and ego syntonic; body syntonic because the child uses his knowledge about his own body movement to instruct the turtle, so that the geometry through movement becomes as much a part of him as the procedures for tying a tie. The learning is ego-syntonic because the programs are coherent with the pupil's intentions: he wanted to instruct the turtle to draw a circle, to see what would happen if ... . Thus Papert would claim his viewpoint of mathematics learning was ideas-centred rather than following a facts-skill centred approach.

Although geometric examples show Papert at his most persuasive, he has extended and demonstrated the use of LOGO more widely (Papert and Solomon 1972), for example in physics, in music and in poetry/grammar. Discussion of this work can focus on Papert's perceptions of teaching and the extent to which the characteristics of LOGO allow these objectives to be accomplished. However, although illustration and case-studies have been provided, there is a lack of detailed recorded evidence on the use of LOGO in the working classroom and an absence of studies which show how pupils' cognitive processing has been influenced and their mathematical thinking improved by using the language. Two preliminary though extensive experiments have been undertaken at a school in Leeds using pupils aged 11 - 13 years studying topics in directed number and in fractions. Somewhat at variance with Papert's intentions, LOGO was used to assist the learning within the conventional mathematics curriculum. A laboratory of five printing terminals was set up and used in an annexe attached to an open-plan classroom. In the experiment covering topics in fractions (their definitions, equivalent fractions and operations on fractions), particular attention was paid to the representations which LOGO could give to fractions, for this determines the types of activity the learner can develop. Fractions can be represented in a numeric, set-theoretic or a geometric style. The one chosen was derived from the last two, with the 'wholes' being collections of elements which could (perhaps) be shared equally. Thus the fraction 3/5 consists of three portions of an equal sharing into five. The elements could also stand as labels for collections made of sub-elements and so the notion of equivalent fraction could be developed. Hence collections of

elements could be built up, amalgamated, shared and transformed. These actions are readily available in the pupils' experience and so could be used to guide their programming activity. The basic LOGO commands were used to define procedures which the pupils used as their own command language; in this way LOGO was made more applicable for the particular subject area. As well as helping the understanding of mathematical concepts, it was hypothesised that the activities of specifying sequences of commands and debugging the program would contribute to the children's ability to analyse and give clear descriptions of the tasks undertaken.

Two treatment groups were used in the experiment, one class following the SMP curriculum option, the other, of equivalent ability, using the language LOGO. The time spent in teaching and study, the size of the teaching groups and the content sequence were equalised for the two treatments, and in an effort to standardise and accurately describe the teaching approaches (so that the experiment could be replicated) a repertoire of teaching modes was defined and adhered to. These detailed the permitted activities of the teacher and of the pupil, the type of feedback and the ways it would be administered, and the decision rules used to complete the teaching mode. Examples of these instructional styles are exposition, demonstration, building (using a pupil's own contribution), problem solving, program formulation, program testing, group exercises. All printouts of work at the terminal and all worksheets and tests used in the classrooms were collected from both groups.

The two classes of thirty pupils were each given pre-tests in number operations, in relational thinking and in spatial reasoning. On all these measures the two groups were equivalent. Both classes had received preliminary training in using LOGO, and for the LOGO treatment the class was divided into groups of 10 so that the main instruction could take place round the terminals. The same teacher taught both classes. The material was extensive in its coverage and the experiments lasted approximately ten weeks.

After the teaching was completed a series of post-tests was administered. These pencil and paper group tests included questions based on the actual material used in the lessons, and others which required application and problem solving in new contexts. A random 50 per cent sample from both

groups was given individual interviews. Questions were similar to those of the post-test but pupils were asked to explain their goals and methods. To measure clarity in expression, the children had to instruct the experimenter how to operate with fractions, and, in more general contexts, how to draw a figure (an arrow in an angled position on squared paper) and how to get to the Headmaster's study from the interview room.

It is difficult to summarise the results briefly and yet do justice to the wealth of data. The LOGO group proved significantly the superior on all the post-test measures of achievement. They were able to express themselves better in the interviews and their instructions were more accurate, explicit and more structured. It was interesting to note that whereas the score distributions for the SMP group were unimodal with a slight positive skew, the LOGO treatment produced scores of greater variance which in some cases followed a bimodal distribution. In short, some of the pupils found LOGO less rewarding even though demonstration procedures/programs were written for the less able to use. A factor analysis of the pre-test scores produced a large general factor, and these factor scores were a good predictor of poor post-test achievement particularly with the LOGO group.

The experiment provides support for the potential benefits of using learning laboratories based on carefully designed educational programming languages. However, some of the less able pupils can find the techniques difficult, classroom organisation with only a small number of terminals is not easy, and teachers need a thorough knowledge and working experience of the language.

During the last few years the Learning Research Group of Xerox at Palo Alto, California, has also developed an educational programming language, SMALLTALK (Goldberg 1979). Although its specification is for teaching design skills, its potential application is wider than that. SMALLTALK is conceived as a component of what is termed the DYNABOOK personal computing system. This is visualised as a portable computing facility the size of a notebook; at present the hardware realisation is a 64K, 16 bit micro-programmable system with high revolution display screen, disc drive, standard typewriter keyboard, and a (pointing) device which controls a cursor on the screen.

2.3 Learner controlled learning

Whilst accepting the value of developing educational programming languages, their functions are limited. For many applications, in order to help students acquire and comprehend the related concepts which define a topic area, programs which can display information and engage the student in question-answer dialogues are necessary. The methods of arranging and sequencing such instruction has been a continual source of debate. In the early 1960's Gagné proposed his hierarchical categories of intellectual skills ranging from discrimination and concept learning to a capability of using higher level rules in problem solving. He proposed techniques for the analysis of educational tasks in order to reveal the knowledge and operations they involve and their relation to this hierarchy of intellectual skills. The hierarchy was to be used to suggest an ordering for instructional sequences, and to provide a framework for developing teaching strategies appropriate to the various types of intellectual skills. As a demonstration of the validity of their ideas, Gagné and Paradise (1961) provided a teaching sequence for topics in probability theory. At lower and initial levels of the hierarchy, mathematical aptitude tests gave an adequate prediction of student performances, but at later and higher levels of the teaching, the best predictor was competence on the earlier learning sequences. More specifically when a pupil was unsuccessful on a lower level and then unsuccessful on a higher module, this dependence was considered to give support for a learning hierarchy. By examining such conditional probabilities of performance Gagné believed he was able to validate his theory, which for some workers argued for mastery level approach to the control of learning. Others have been more critical, questioning whether the hierarchies are logical rather than psychological in nature. It has been maintained that students can learn at a variety of levels and that by maintaining a strict performance management of learning there is no exercise of self-evaluation for the student and no effort after learning how to learn.

A large CAL project in the USA, designated TICCIT (time-shared, interactive, computer controlled information television), has implemented a system which allows the student substantial control over the educational decision-making. Although TICCIT has developed interesting and economical hardware, a main feature is the way educational material is designed and

utilised. In a learner-controlled system, the student should be able to (i) set his own learning goals (ii) determine a strategy to realise them and (ii) have the material labelled and available so that the strategy can be used to attain the learning goals. These requirements determine the type of student control language which is developed. The user must be able to survey the course material and the inter-relations between topics sometimes in summary form, sometimes in more detail. In the TICCIT system the topic material is partitioned into rules or explanatory principles or examples. The content can vary on a hard-easy dimension by altering the abstract-concrete terms in which the material is expressed. The teaching mode can be expository (telling) or inquisitory (asking) and the student can ask for help if he feels this to be necessary. This can cause the system to colour-cue and highlight important parts of the text, and the examples of concepts are presented in matched forms, i.e. examples and non-examples are shown together on the display screen. The student commands, in the form of control buttons, are shown in Figure 1.

KEY	*FUNCTION DESCRIPTION*
MAP OBJECTIVE	*Controls Learning SEQUENCE*
RULE EXAMPLE TEST	*Controls Learning MODE*
HARD EASY	*Controls Learning LEVEL*
GO BACK SKIP REPEAT	*Controls Learning PACE*
HELP ADVICE	*Controls USE of Facilities*

FIGURE 1: The TICCIT System Function Keys

Thus the learner himself determines his objectives and decides on the type, sequence and pace of his instruction. To help him make these decisions an elementary Advisor can provide comment on his progress and previous judgements.

During 1975-76 an independent and large-scale evaluation study of TICCIT programs was carried out (Alderman 1978). In the mathematics area, five thousand students were involved covering 200 sections of target courses in American community colleges. All students must have completed at least one term's exposure to TICCIT material before they could be included in the study. The programs served as a direct replacement of teachers so the study compared TICCIT classes with equivalent groups of students following the conventional lecture-discussion methods in the same mathematics courses in the same colleges.

The results were interesting and perhaps unexpected. In the conventional mathematics teaching, a completion rate of 50% was recorded, i.e. half the students satisfactorily received a credit grading at the end of the course. For the TICCIT group the completion dropped to 16%. Attitude questionnaires completed by these students also showed less favourable attitudes to mathematics than did those following the lecture-discussion treatments. (Perhaps it should be noted that the criteria for course completion differed between the treatments. The TICCIT system barred progress at the end of certain sections unless satisfactory performances were recorded. The conventional treatment offered no such hurdles.) However, it appears that TICCIT was producing a detrimental attrition effect which Alderman suggests may be a consequence of the high degree of learner control. "Programs that allow each student to proceed at his or her own pace risk losing students unable to manage their own learning." An analysis of post-test data was undertaken to compare levels of achievement. On tests closely following the material of the course, the TICCIT groups obtained scores which were, on average, more than 10% better than those who had been conventionally taught. Other post-tests based on application and problem-solving tasks showed even greater differences in favour of the TICCIT system.

Some evidence of the effects of learner-controlled systems for teaching topics in Statistics (experimental design) were obtained from some sixty

Psychology students at Leeds University. The undergraduates were randomly assigned to two teaching groups. One followed a program controlled route through the material which was regulated by individual performance levels. The other assembled facts, definitions, examples and test questions in labelled files which the student could access as he wished. Post-test results clearly favoured the program-directed group, though these benefits were largely gained by the students with less mathematics experience. However, questionnaires showed that the undergraduates gave higher ratings of satisfaction and stimulation to the learner controlled materials.

A more cogent comment on learner control has been made by Pask and his co-workers. They have been concerned with the learning styles students adopt when placed in free learning situations, and how these characteristics are associated with differing cognitive structures. Pask distinguishes between what he terms 'serialists' and 'holists'. The former prefer to gather together and relate material through a series of simple linkages. Similarly, their explanations and exposition of topics tend to show the same characteristic structure. In contrast, holists tend to build more complex associations and be more wide-ranging in the material which they bring together into their mental framework. Of course these are predispositions rather than absolutes, and in his latest work Pask has modified his descriptions and prefers the labels of operationalists and comprehensives, with the term versatiles being reserved for those students who can and do operate in either mode. It is also maintained that independent and reliable measures of these mental attributes can be obtained through a series of tests (e.g. the Spy-ring test) which can be administered and marked by computer. Concurrently with this work Pask has developed techniques for analysing subject matter and educational conversation; these were used to design materials for teaching Statistics. The resulting system could be managed by computer and could give a great deal of learner control over content type and its sequence. The materials consisted of teaching segments, information files, an experimental statistical laboratory and self-testing materials. The subject matter structure was displayed for the student so that he could elect to learn sub-topics singly or in combination.. Pask used this system to undertake a series of experiments. For example, he found that when teaching sequences were imposed which were serialist in nature, i.e. interrelated topics ABCD were studied in sequence

such as A, B, then A related to B, C, $\overline{ABC}$, then D, finally $\overline{ABCD}$, they suited the serialists better than the holists. The converse was found for teaching sequences such as A→B and C→D, then $\overline{ABCD}$. This evidence seemed to favour matching teaching styles to the learning dispositions of the student. Pask also found that learners in a free choice situation did not always make these advantageous decisions. Hence, although he advocates initiatives being given to the students, their decisions should be checked or discussed either with a teacher or a controlling program.

3. RULE LEARNING

As well as learning the properties of mathematical concepts, it is necessary to understand the ways in which they can be used. This requires the learning of rules, e.g. rules for evaluating numerical expressions, for carrying out transformations on trigonometric identities, and rules for differentiation and integration. These not only provide useful techniques but illustrate and develop the properties of number systems, algebras and geometries. Thus rule learning requires practice, but what factors should govern its efficient management?

There is evidence from Educational and Experimental Psychology (e.g. Nitsch, 1977) that same-context types of practice increase remembering but do not aid transfer. Varied-context examples, however, aid transfer but are liable to cause confusion in the initial stages of learning. Thus a practice menu which follows a hybrid procedure is likely to facilitate the ease and speed of initial learning and also aid flexible transfer on subsequent tasks.

A well documented principle, and one which is important for the management of practice, is the influence of feedback (i.e. the message following the student's response). Many of the early studies using programmed texts failed to show any benefits from providing knowledge of results; indeed, in a review of thirty-five studies (Grundin, 1969), of which thirteen were specifically testing the influence of feedback, not one showed clear and significant differences in its favour. Anderson et al (1971, 1972) in a series of computer based experiments gave some answers to this puzzle. He showed that it was an over-simplification to consider

knowledge of results as a reinforcement which strengthened the association between the student's response and the stimulus material to which it was made. In the human learning situation the more important influence of feedback is to locate error, and provide information to enable the student to correct it. Thus, rather than designing small step practice programs which guarantee an extremely high level of success, the task difficulty should be matched to student competence so that his misconceptions are revealed. Then appropriate feedback and supporting practice can be given.

These techniques require knowledge of the ways learners undertake the tasks - the operations and information they use or misuse in their work. Ideally this knowledge should not be merely in the form of static performance and error records, but as procedures which are models of the student's cognitive processing. If it isn't, then prediction is either difficult or overgeneralised.

Arithmetic practice with young children is a topic which has attracted a considerable amount of computer based work, for example the pioneering experiments of Suppes (1967) in California, whose materials have been adopted in a variety of school districts in America. Over the last few years the Computer Based Learning Project at Leeds has also undertaken research in this area. In an attempt to derive models of task difficulty, a series of printed practice exercises were given without feedback to several hundred schoolchildren aged seven to eight years. For this age group, simple counting models for vertical addition and subtraction were proposed. These were expressed as flow diagrams so that for any example/task, the number of times various operations were used could be estimated. By attaching error values to each operation, the probability of successfully completing the example could be computed in terms of these parameters. From the performance data, following a least squares fitting analysis, estimates of the parameters of the model were obtained (Woods and Hartley, 1971). These models were then used by computer programs to generate tasks of specified difficulty when the pupil was at the terminal. From his performance records, individual parameter values were estimated so that the model became a representation of the processing of that particular pupil. An experiment was then conducted in which the computer programs generated examples to maintain a working level of success for the pupil of either 90% or 75% or 60%. The style of feedback was kept constant

between the three treatments to which approximately ninety pupils were allocated on a random basis. Each learning session was timed for twelve minutes with equal numbers of pupils from each treatment group working individually and concurrently at the printing terminal. Twelve practice sessions were allowed for each pupil. Pencil-and-paper pre- and post-tests were given. The results were clear. They favoured the immediate difficulty (75%) group, followed by those with a high working level of success. The treatment with a low success rate was easily the worst and was not liked by the pupils.

If task difficulty can be regulated from information held about the pupil's processing, so can the appropriateness of the feedback message and the type of remedial practice which is given to remove persistent errors. In a second experiment to demonstrate the effects of different types of feedback, the initial results in arithmetic tasks were disappointing and no significant improvements in performance were recorded. Observation of pupils showed the reason why. The program evaluated each digit the pupil typed. If correct it was printed, if not a bell rang and the pupil tried again. If this second attempt were wrong, the feedback message was then printed and the program immediately proceeded with the next question. In other words, the program did not require the pupil to read, much less understand the feedback message. When these controls were placed in the program, the results were very different. Contrasted with a control group - who had practice sessions at the computer but who received no feedback except the number of examples they had answered correctly - two other groups received more informative feedback. In one (labelled passive mode) the method for working out the correct answer was printed, after which the pupil had to copy and type in the correct answer. The other 'active' mode group received the same feedback message, but divided into those question-steps the pupil should have been asking himself. To each step the pupil had to supply the correct answer. If incorrect, the right response was printed and the same question step repeated. On completion of the feedback, the same example was printed up to the point of error, after which the pupil continued in his working. This active mode of supplying feedback took more time, but allowed greater control over the student's processing of the given information. The type of tasks featured in this experiment were multiplication of up to three digit numbers by a single digit number, and

the menu of tasks, which was identical for each treatment group, was carefully balanced for item difficulty. The CAL teaching was arranged into twelve minute working sessions. Approximately ninety pupils aged nine years took part in the experiment. They were given pre-tests on knowledge of tables and multiplication tasks, then divided between the three treatment groups following a stratified random sampling procedure. After the teaching of twelve working sessions, pencil and paper post-tests were administered.

The control group made only slight learning gains and was significantly worse than either of the other two groups. Although the additional time taken up with the administration of active feedback meant that this group had only half as many examples as the passive feedback group, their performances overall were equivalent and both treatments showed substantial learning gains (Tait et al, 1973). A regression analysis of post-test performance against mathematical competence (taken from pre-test results) revealed that the less able benefitted much more from the active feedback condition. Their performances were significantly better than the performances of children of similar ability in the passive feedback group. For the more able children the treatments were more or less equivalent. Thus the results argue for an adaptive decision rule in which the type of feedback is related to the competence of the child.

In rule-learning the research emphasis has been placed on showing the effects of a discriminatory selection of tasks (in difficulty and in same/varied context) and of feedback (in the particular forms of its message and the terms of its application). To use these factors depends on knowledge the program holds about the task and about the student In order to be adaptive the program must be able to carry out the task in ways which mimic those of the student and those of the expert. Following comparisions between the two, decision rules can operate the generation or choice of tasks, and the type of feedback which is given.

A criticism sometimes made of practice tasks is that they do not stimulate the child and can lead to a routine and dreary execution. The computer, however, can formulate these exercises as games - though this is a technique which at present is not well exploited by CAL. However, Burton&Brown (1979) have developed a computer board game similar to arithmetical snakes

and ladders. Each player receives three numbers (from spinners or from a random number generator) which he can use to form any arithmetic expression (using the operations +, -, x and ÷ as well as parentheses). When this expression is evaluated, it determines the player's move, and the aim is to complete a course of seventy spaces before any opponent. The best strategy is not necessarily to produce the largest number as there are short-cuts - 'ladders' - which the player is allowed to find by moving forwards or by moving backwards to a lower position. Further, if he lands on a 'town' square (which occurs every ten spaces), he can move on to the next one; if he lands on an opponent's square ('bumping'), that competitor is moved back two towns. Of course the program is more than a game; it is a teaching device designed to illustrate the four arithmetical operations and the use of brackets.

The teaching components have to operate at several levels. First, there is the forming and evaluation of the arithmetical expressions; second, the determination of strategy (e.g. moving forwards or backwards to obtain a town, a shortcut or a bumping); and third, the aspects of game skill (e.g. keeping an eye on opponents' positions and considering what options they can follow). Within each level, issues such as those given above are specified. Note that the program can monitor the pupil in these terms (i.e. work as the pupil) and also operate in expert mode. It is the dialogue between these two projections of play which determines the teaching. For example, at the first level the dialogue may consider the pupil's use of brackets - were these necessary, or was the expression correctly evaluated? The second level would have dialogue on possible moves and their relation to an expert as well as to a pupil's strategy. Of course, deciding when the expert program should interrupt the player's game, what particular issues should be discussed, and the nature of feedback, explanation and help are pedagogical matters. Brown has set out the pragmatic principles he followed in some detail, and the overall structure of the program is compact and hierarchical. The general point is that these practice programs are beginning to work in a knowledgeable manner somewhat akin to the human teacher. They have knowledge of the task for they can generate and solve practice examples in the role of an expert; they also have multi-levelled knowledge of the students' operations and strategies. The decision rules which use this information to provide and to control tasks

and feedback encapsulates the program's knowledge of teaching.

It is a mistake to consider that intelligent teaching systems can only deal with number and simple arithmetical tasks. Research at Leeds has developed a program which can be used to study practice in several mathematical domains. The system operates through transformation rules and can be used for Boolean Algebra, Trigonometry and Algebraic Simplification with little alteration to its source coding. At present the main purpose is to study the types of rules which pupils use and the errors they make when simplifying algebraic expressions. In one version of the program, the student successively chooses one of a set of rules to apply and so works out his solution steps. Since the program itself can generate problems, and work them out in expert mode, feedback can be provided for two main purposes. The first answers the questions: Can the selected rule be legitimately applied? If not, why not? The second considers the o[illegible]: If the rule can be applied, is it the most efficient one to use to [illegible] solution? To answer this latter question requires knowledge of sub[illegible] which should/could be set at that point in the working, and the progre[illegible]e rule will make down the solution tree. Following this analysis by the program, feedback can be given. If a pupil is unable to proceed, the program can help by working out those rules which can be applied. Then, if further help is requested, the program gives a hint as to which one(s) is/are the most efficient. This help has to be purchased by the student, i.e. he forfeits a certain number of points. At the end of the question the computer prints its solution for comparison.

The program keeps detailed records and makes inferences about each student's working methods and the causes of his errors. These relate the characteristics of the algebraic expressions to the rules the student applies or misapplies. Currently such descriptions are being compared with error analyses produced by teachers giving individual diagnostic interviews, on similar tasks, to the same pupils.

All these programs, when operating in a teaching mode, use decision rules which reflect the author's pedagogical assumptions. These should relate to factors known or thought to influence learning or cognition, and should be validated by experiment. Unfortunately, although these CAL programs provide excellent laboratories for such studies, there is a lack of experimental data. O'Shea (1979), when conducting research at Leeds, had the idea of making the program itself undertake the experiments. It was

called a Self-Improving Teaching System, for it was able to alter (i.e. reprogram) its teaching strategy at runtime on the basis of evaluative data it collected. The subject area was teaching the solution of quadratic equations and the program, in common with the other examples outlined above, was given a hierarchical structure. First, there was the teaching within the problem task, i.e. monitoring and evaluating pupils' responses and providing help and feedback. Second, there was a lesson controller, which, on the basis of feedback data passed to it from the first level teacher, managed a menu of tasks. The highest level of the program functioned as an educational psychologist. It was given high level goals such as minimise practice time, reduce error rate, increase performance gradient, and it was able to make assertions relating task type, sequence, and type of feedback, to how well they would promote these goals. The program was then able to devise the experimental treatments and set up task menus, with appropriate feedback instructions. Pupils coming to the terminal were randomly allocated to these treatments. From the data a t-test analysis was carried out by the program to evaluate its assertions. The program also had some primitive deductive capability so that it could alter related assertions without further experiment and also interconnect the results from a series of experiments. The system had some success when used with schoolchildren and showed that it could become experienced and improve its teaching performance.

Sometimes teaching objectives require students to make conjectures about relationships between variables and the functional forms which can be used to express them. For these purposes CAL simulation programs have been developed particularly for learning mathematical applications to the physical and social sciences. Typically in simulation exercises, the user can provide input values for labelled variables. Through the output displays which the program computes, and through systematic trials, the learner can appreciate and perhaps even specify the form of the underlying function or model. However it should be noted that the programs theselves provide no direct teaching; in general they give no guidance on strategies for the systematic collection of data, or methods for its interpretation. Further, no individual records are kept, so the programs are not able to relate the learner's previous experience to his present methods of working. Moreover

it is unusual for the student to be able to alter or edit the model on which the simulation is based, or directly insert competing models.

However, Dorn (1977) has provided an illustration of how such mathematical modelling methods can be used in the social sciences. A feature of his work (which was implemented on microcomputers) was allowing mathematical reasoning, and testing through computional models, to complement each other in the learning process. The computer was a laboratory which provided computational tests for mathematical reasoning; the results in turn provided data to stimulate further mathematical exploration, and so on. Dorn took a series of examples including population growth, predator-prey and epidemic models. These all involved difference equations which in general have no analytic solution. For example, a simple model of the growth of a single species in a closed environment has a constant growth rate given by:

$N_{k+1} = (1 + A)N_k$ where N_{k+1} is the population after time interval $(k + 1)$, and A is the growth rate (e.g. births less deaths). The equation sets have solution $N_k = (1 + A)^k N_0$ where N_0 is the initial population. If $A > 0$ then $(1 + A)^k \to$ infinity, as $k \to$ infinity; this model must be rejected as the population grows without bound. Hence the requirement is for a retardation when the population becomes large. Perhaps $N_{k+1} = (1 + A - BN_k)N_k$ may be suggested. A program can be used to evaluate this for a series of time values, and allow the learner to input his values for A and B. But what are the properties of this solution - when B is held constant at a small value 0.0001 (say)? The behaviour of the model turns out to be interesting for when A = 4 the population becomes extinct after 4 time intervals. From experiment the data shows that oscillations start at about A = 1.5 and become so violent that when approaching A = 4 the population vanishes. So what is the largest value of

A (the growth rate) for which no oscillations occur? If the population does settle down, to what value of N_k does it tend?

Mathematically N_{k+1} must equal N_k = N? (say) for stability. From the equation above, N? must either equal zero or:

$1 + A - BN? = 1$ i.e. $A = BN?$ i.e. $N? = A/B$

which for A = 0.5 and B = 0.0001 means N? = 5000.

But under what circumstances is a population going to be stable and approach this equilibrium value? The computational laboratory can be used to answer this question by allowing N_0 to equal A/B + a small constant (perturbation) - and so the argument can proceed.

Hopefully this gives the flavour of Dorn's approach which requires students to program but in a way which is used to illustrate mathematical properties of sets of equations; with mathematics being used to determine the form of the programs. This laboratory approach has much in common with the way teachers and mathematicians operate. Variables of interest are proposed and labelled , functions/equations set down, consequences explored and matched with data taken from experience. Whether BASIC - the language Dorn used - is pedagogically appropriate, or whether it is better to develop specialist subsets of commands and procedures which fit more easily into the educational task is a matter for debate and experiment.

4. PROBLEM SOLVING

In these latter rule-learning applications, the learning techniques have moved well away from algorithmic types of practice. There are several, even many legitimate models which may be proposed to explain variations in the data. Indeed these learning tasks have elements of problem solving where the student works out how to move from an initial to a final goal state and is expected to be able to give some explanation and justification

for his decisions.

Usually problem solving demands the application of concepts and rules which are known by the student, to situations which are, in some sense, novel. Such activities have long been studied by psychologists and their particular views have influenced the types of tasks they have chosen for their experiments, the ways they have described problem solving performances and the conclusions they have drawn. To Gagné, problem solving was a process of applying higher order rules, which have to be learned in an orderly and hierarchical fashion. To gestalt psychologists it was a matter of perceptual and cognitive organisation; to practitioners such as Polya the feature of interest was the heuristics which are developed to control goal-setting, strategy and evaluation. However, the approach which is most closely related to computer simulation and modelling is that of information-processing, which is followed by psychologists such as Newell and Simon (1972). Over a period of twenty years they have undertaken an extensive series of studies. A first development was a computer program (GPS) which could solve a variety of problems when these were expressed in terms of operations transforming symbol strings. From this, and other experimental experience, Newell and Simon have set out their descriptions of problem solving performances which consider the human solver as an information processing system. They are interested in the process of problem perception, i.e. in the techniques of restatement and elaboration which subjects undertake as they set up the goals and givens in more experience-related and, perhaps, more formal terms. The strategies for achieving sub-goals and the ways in which these are worked out and their progress evaluated, is a second theme of interest. Finally there are the conclusions the subject makes in interpreting his answer - its areas of application and dependence on simplifying assumptions. So the descriptions of problem solving are detailed and individual differences a recurring theme.

The information processing system, which represents the problem solver, consists of effectors and receptors, a processor and a long term memory. The memory contains semantic information in the form of symbol structures, i.e. symbol elements (e.g. digits) and the operations which can be performed on them (e.g. rules of arithmetic) and the relations which exist between them (e.g. equality, containment or logical dependence). The

elementary information processes (eip's) which operate on the symbol structures do so in the processor, which has limited capacity. The eip's must be comprehensive enough to encompass the behaviours of subjects, for sequences of eip's will be the programs which must replicate and hence describe problem solving performance. Newell and Simon recommend that these programs, which encapsulate the problem solver's strategies, should be set down as production systems i.e. as sets of condition-action statements. The action is an eip or an eip set and the attached condition necessarily causes the particular eip to operate. The conditions are expressed as tests of value-sets attaching to those symbol structures which are the subject's knowledge of the problem. After its operation the eip alters the contents of the processor, and which eip then operates is the one whose conditions match the (new) values. Thus the production system of the subject (which represents his strategy) and the contents of memory will be used to describe his problem solving performance and his current problem solving capabilities.

Newell and Simon believe that a first requirement for research in problem solving is a reliable technique for describing performance and they have given a great deal of attention to techniques for transforming students talk-aloud protocols into the production system format outlined above. The advantages of putting descriptions in terms of computer programs is the precision and consistency they require of the experimenter, the framework they provide for evaluating individual differences and their use in predicting performance on related problems. The programs provide dynamic models and records of a subject's behaviour. When solving the problem he is asked to talk fully and continuously to reveal his aims, methods and difficulties. This protocol is transcribed and segmented into statements which show the steps of his progress, and into paragraphs which show a sequence of steps under the action of a goal such as 'identification of variables'. From this, the first component of the description is constructed - a problem behaviour graph. This is set out as a series of knowledge states through which it is supposed the subject has passed at successive steps of his working, and as operators which act on these knowledge states. Thus a knowledge state can be thought of as a node on the problem behaviour graph, and the arc which connects adjacent nodes is the operator transforming the input (the entry knowledge state) to the state

shown in the output node. The graph is linearly ordered by time of generation, but the subject may backtrack and return to a previous knowledge state.

A second component of problem solving descriptions is the problem space. This is a specification of the symbol elements, their relations, and all the operations acting on them, which are associated with the problem. In short, it represents a composite of possible routes which experts and other subjects might take in their step-by-step solutions. It also includes general problem solving heuristics. The problem solving descriptions are completed by two dynamic components, namely the production system (which represents the strategy followed by an individual) and the problem solving trace, which is the printout trace of steps when the production system program is applied to the problem space. Correlation between the trace and the subject's problem behaviour graph is evidence of the internal validity of the production system.

Whether human thinking and problem solving can be usefully represented by such deterministic and reductionist schemes has been much discussed. The technique is certainly time-consuming and difficult, particularly with less able pupils who tend to behave in a disjointed and inconsistent fashion and who are not clear in their verbal descriptions. Newell and Simon have limited their approach to problems in cryptarithmetic, logic and chess; others, for example Klahr and Wallace, and Young, have used the techniques with young children engaged in Piagetian-type tasks of conservation and class inclusion. At Leeds we have undertaken some preliminary work in studying how first year sixth formers use mathematics in solving mechanics problems. Experimental variables were problem difficulty and context - which involved either fixed or moving reference frames (e.g. a moving/accelerating lift). The problems used Newton's laws of motion and/or equations of uniformly accelerated motion. The aims were to study individual differences between students, to consider their problem solving strategies (production systems) and determine to what extent they altered when problem context and difficulty were changed. Twelve students took part in the study and were graded for competence on the basis of their expected A level grades in mathematics and the sciences. A pre-test ensured that all subjects had the necessary knowledge of physics and mathematical techniques which

would be needed for solutions.

The twelve students were individually interviewed and asked to give talk-aloud protocols as they worked through five problems. All statements were tape recorded and transcribed. From these protocols, problem behaviour graphs and production system descriptions of the strategies were constructed.

The results showed considerable individual differences in standards of performance. For example four of the twelve students were successful on all problems, but one subject had none correct. The time spent answering the problems varied from 68 to 210 minutes, and the average number of statements/steps used by a student in answering each problem ranged from 6 to 19. All these variables on accuracy of solution, time spent and length of solution correlated highly in expected directions with problem difficulty and the students' mathematics/science ability gradings.

Although the protocols and problem behaviour graphs revealed particular errors, the main conclusion was the rigidity which students showed in their approach. There was little change in an individual's problem solving method even when changes in problem contexts meant that the method was not appropriate. Overall there was only small alteration in a subject's production system across problems. Those who were the more competent problem solvers tended to have a certain type of production system; they collected and recalled all relevant information so that they were then enabled to apply a 'strong' production, i.e. one in which there were specific and detailed conditions attaching to the operations. Those who were less competent tended to close prematurely on a less strong production, i.e. they were less discriminating. Also they seemed unable to backtrack clearly, and seemed confused rather than helped by information they were producing as they worked at the problem. In short, their problem formulation appeared weak, with mechnanistic application of formulae without an adequate working-out of the physics of the situation - hence self evaluation was poor.

So, in summary, the technique of production system analysis, while having potential, requires much more work and wider application before expectations can be fulfilled. Within CAL, the longer term aim is to use these techniques to model students' processing and to form a dynamic record of their working which would be used in teaching programs to guide decision-making, dialogue on strategy, and the help/feedback given to the students.

A second important aim of the talk-aloud protocol and its analysis was to make transparent the cognitive processing undertaken by the student. This information may then be used to underpin the design of an educational programming language suitable for that topic area. Indeed this feature of making a student's working transparent and so open to discussion is one of the advantages quoted for LOGO. In using the language, the learner instructs the program how to undertake the solution of the problem - in effect, he writes his production system. Hence teaching with LOGO should be cast in a problem solving mould.

In teaching problem solving considerable initiative has to be given to the learner so that he can develop his own methods and the educational discussion between student and program/teacher has to proceed at three levels. At an intermediate level is discussion of goals (Why?) and methods (How?). Within this framework, on a lower stage, are the responses which make up the problem solving, and their evaluation and feedback. There should also be discussion about problem characteristics, choice of problem and learning strategy, so that experience can be generalised and the student can learn how to learn. This is the third level of dialogue. During the last two years we have been developing such CAL modules for teaching the application of mathematics to the biological sciences. The topics include using logarithmic and exponential functions, and the elements of calculus. The CAL materials are pre-stored but provide the student with a command language so that as well as typing responses in answer to problem he can ask about the program's evaluation of his errors. Equally important the commands allow him to select the type and size of the problem solving task, to ask for help on goals (Why?) and methods (How?) and to have access to supporting definitions, facts, rules and a calculating facility. Further commands allow him to retrace steps and have a review or summary of progress. (A synopsis of these commands is given as an Appendix.)

In addition to the problem solving modules, there are materials which teach and practice the basic mathematics directly. Full details of performances on the programs and all the types of commands used by the student are stored. An Advisor program is being designed to give discussion on learning goals, on choice of material and on methods of working. In this way

it is hoped the student will learn to evaluate both his progress and his learning methods. Preliminary experience with these materials has proved encouraging. However, it is clear that CAL programs will require more knowledge of the problem solving domains, of ways of representing student strategies and of how they are developed, before the teaching will be sufficiently adaptive and intelligent in its advice giving and decision making.

5. CONCLUSIONS

An important feature of CAL materials is that through their designs they can attempt to relate theories of learning, cognition or problem solving to educational practice. But there is a considerable difference between casting programs for a laboratory type experiment and undertaking an educational development which will produce sufficient material and acceptance to make a significant impact on teaching method. Integrating individualised CAL methods within the conventional teaching classroom will not be easy, and it is likely to take several years' experience before a teacher learns to make most use of them. The traditional methods of training teachers, both in subject matter expertise and teaching practice, is designed to enable them to develop their own teaching initiatives. Similarly, CAL materials must not be tightly constrained, but allow the teacher to use and develop them in accordance with his own educational intentions and the specific requirements of his classes.

Our experience in trying to deal with these problems has resulted in the production of high level languages which allow teachers themselves to design CAL material. For example, in the problem solving modules for mathematics applications for the biological sciences, pilot modules were written in close consultation with teachers. Then students tried them out and after several amendments the style of the programs and the command language were considered to be satisfactory. High level translation programs were then implemented so that if teachers set down, in standard form, the problems, the steps and stages of their solution, together with the 'How' and 'Why' explanations, generated codes in the base author language would be produced. The programs are automatically tested and parsed into the run-time program for

use by students. Again, for the arithmetic practice program referred to earlier, the teachers themselves set up a library of example types by specifying the parameters which control the question generation and type of feedback. From an amalgamation of example types, lessons can be composed. The sets of lessons form a curriculum which is entered against the pupil name/number of all intended users of that curriculum. In the Statistics materials, a high level language has been implemented which allows teachers to select and sequence small pre-written teaching modules, and to connect these with their own explanatory comments and decision rules. To this controlling program is attached the corresponding class-list which contains the names/identifiers of its students. Hence their instruction is individually managed. Records are kept automatically, and an evaluation language has been designed so that teachers can interrogate these records and have the data processed in ways which they choose.

So, what of the future? CAL is providing many ideas to stimulate and inform, but there is a lack of systematic research and evaluative experiment. In particular, there is a need for research studies into the process of cognition, learning and problem solving within mathematics in terms which relate to the computer medium. Knowing of factors which influence learning gives suggestions or starting points for designing teaching strategies which control presentation variables and instructions/tasks for the student. These principles need to be systematically tested by using the program designs under laboratory conditions, so that individual differences and interactions between treatment and student variables can be closely studied. In order to achieve the long-term aim of assimilation within conventional teaching, it is essential that CAL experience is documented in a systematic manner, and that evaluation studies, including those of the 'illuminative' case-study type are carried out. Designing languages for both teachers and learners requires considerable expertise, but techniques from Artificial Intelligence and Cognitive Science can provide useful guide-lines.

During the next few years, developments in microcomputers and their associated use with larger machines will bring the techniques of Computer Assisted Learning to more prominence in both Secondary and Higher Education. The problems in realising this potential for learning and teaching will require co-operation and co-ordinated efforts from teachers and technical working groups if benefits are to be enjoyed by students - and that after all is our common concern.

REFERENCES

ALDERMAN, Donald L. (1978). Evaluation of the TICCIT Computer-Assisted Instructional System in the Community College: FINAL REPORT. Educational Testing Service, Princeton, New Jersey 08541, USA.

ANDERSON, R. C., KULHAVY, R. W., and ANDRE, T. (1971). Feedback procedures in programed instruction. *Journal of Educational Psychology*, 62, pp.148-156.

ANDERSON, R. C., KULHAVY, R. W., and ANDRE, T. (1972). Conditions under which feedback facilitates learning from programed lessons. *Journal of Educational Psychology*, 63, pp.186-188.

BURTON, R. R., and BROWN, J. S. (1979). An investigation of computer coaching for informal learning activities. *International Journal of Man-Machine Studies*, 11, 1, pp.5-24.

CRAIK, F. I. M., and LOCKHART, R. S. (1972). Levels of processing: A framework for memory research. *Journal of Verbal Learning and Verbal Behavior*, 11, pp.671-684.

DORN, William S. (1977). Modelling and Computing in the Social-Behavioral Sciences in Computer-Based Science Instruction [JONES, A. and WEINSTOCK, H., Editors] pp.141-184. Noordhoff-Leyden.

EYSENCK, Michael W. (1978). Levels of processing: A critique. *British Journal of Psychology*, 69, pp.157-169.

FREDERIKSEN, Carl H. (1975). Acquisition of semantic information from discourse: Effect of repeated exposures. *Journal of Verbal Learning and Verbal Behavior*, 14, pp.158-169.

GAGNÉ, R. M., and PARADISE, N. E. (1961). Abilities and learning sets in knowledge acquisition. *Psychological Monographs*, No.518.

GENTNER, Donald R. (1976). The structure and recall of narrative prose. *Journal of Verbal Learning and Verbal Behavior*, 15, pp.411-418.

GOLDBERG, Adele (1979). Educational uses of a Dynabook. *Computers and Education*, 3, 4, pp.247-266.

GRUNDIN, H. U. (1969). Response mode and information about correct answers in programmed instruction. In, *Aspects of Educational Technology* [MANN, A. P., and BRUNSTROM, C. K., Editors], Vol.III. Pitman.

MAYER, R. E., and GREENO, J. G. (1972). Structural differences between learning outcomes produced by different instructional methods. *Journal of Educational Psychology*, 63, pp.165-173.

NEWELL, A., and SIMON, H. A. (1972). *Human Problem-Solving*. Prentice-Hall.

NITSCH, K. E. (1977). Structuring decontextualized forms of knowledge. Ph.D. dissertation, Vanderbilt University, Nashville, Tennessee, USA.

PAPERT, S. (1971a). Teaching children thinking. LOGO Memo No.2. Artificial Intelligence Laboratory, Massachusetts Institute of Technology, USA.

PAPERT, S. (1971b). Teaching children to be mathematicians *v.* teaching about mathematics. LOGO Memo No.4. Artificial Intelligence Laboratory, Massachusetts Institute of Technology, USA.

PAPERT, S., and SOLOMON, C. (1972). Twenty things to do with a computer. *Educational Technology*, 12, pp.9-18.

SUPPES, P. (1967). Some theoretical models for mathematics learning. *Journal of Research and Development in Education*, 1, pp.5-22.

TAIT, K., HARTLEY, J. R., and ANDERSON, R. C. (1973). Feedback procedures in Computer Assisted Arithmetic Instruction. *British Journal of Educational Psychology*, 43, pp.161-171.

WOODS, P., and HARTLEY, J. R. (1971). Some learning models for arithmetic tasks and their use in computer-based learning. *British Journal of Educational Psychology*, 41, 1, pp.35-48.

APPENDIX: Student Commands for Controlling Teaching

Whenever < > is printed, the following commands are valid:

(1) To regulate size of step in working out the problem:

ANSWER← (returns to) requiring an answer to the problem itself

STAGES← allows the question to be answered in stages.

STEPS← takes you through the question step-by-step.

(2) For help on goals and strategy:

HOW← gives advice on how to accomplish the last-mentioned task.

WHY← gives a reason for attempting the current task.

HOW LAST← returns to the previous task and gives advice.

WHY LAST← returns to the previous task and gives a reason.

(3) For summarising and regulating the route through the problem:

QUESTION← prints the text of the current question you selected.

REQUIRED← prints the text of the question/stage/step to which an answer is currently required.

LAST← returns to the previous statement.

START QUESTION← starts the question afresh.

START STAGE← starts the current stage afresh.

START STEP← starts the current step afresh.

REMIND← prints a summary of how you got to this point.

REVIEW← prints a summary only of questions you received and answers you have given.

STOP← returns to **!

(4) Obtaining information and feedback:

After an error of a particular type

WHY WRONG← will give further information.

TELL← tells the answer to the question

<word or phrase>#← gives a definition of word or phrase. *(If available.)*

<word or phrase>?← gives information. *(If available.)*

When

<*> is printed, either

SHOW← can be typed to obtain a summary of the method of the last stage or the whole question, whichever is appropriate.

Or

← can be typed to move on to the next stage or question.